**Proiect Sisteme de Gestiune**

**a Bazelor de Date**

**-baza de date pentru un lant de restaurante-**

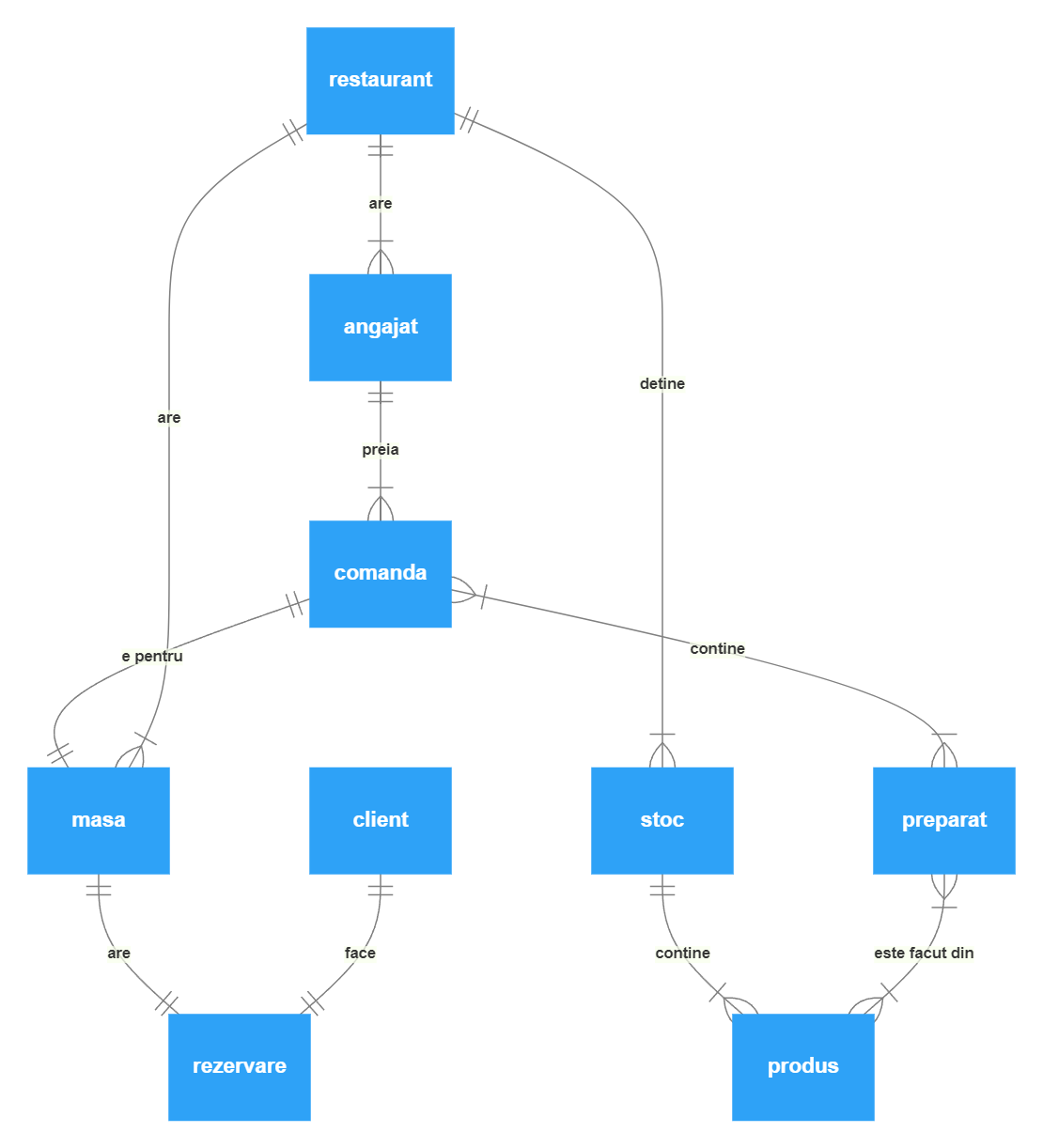
**Botezatu Cosmin-Adrian**

**Grupa 242**

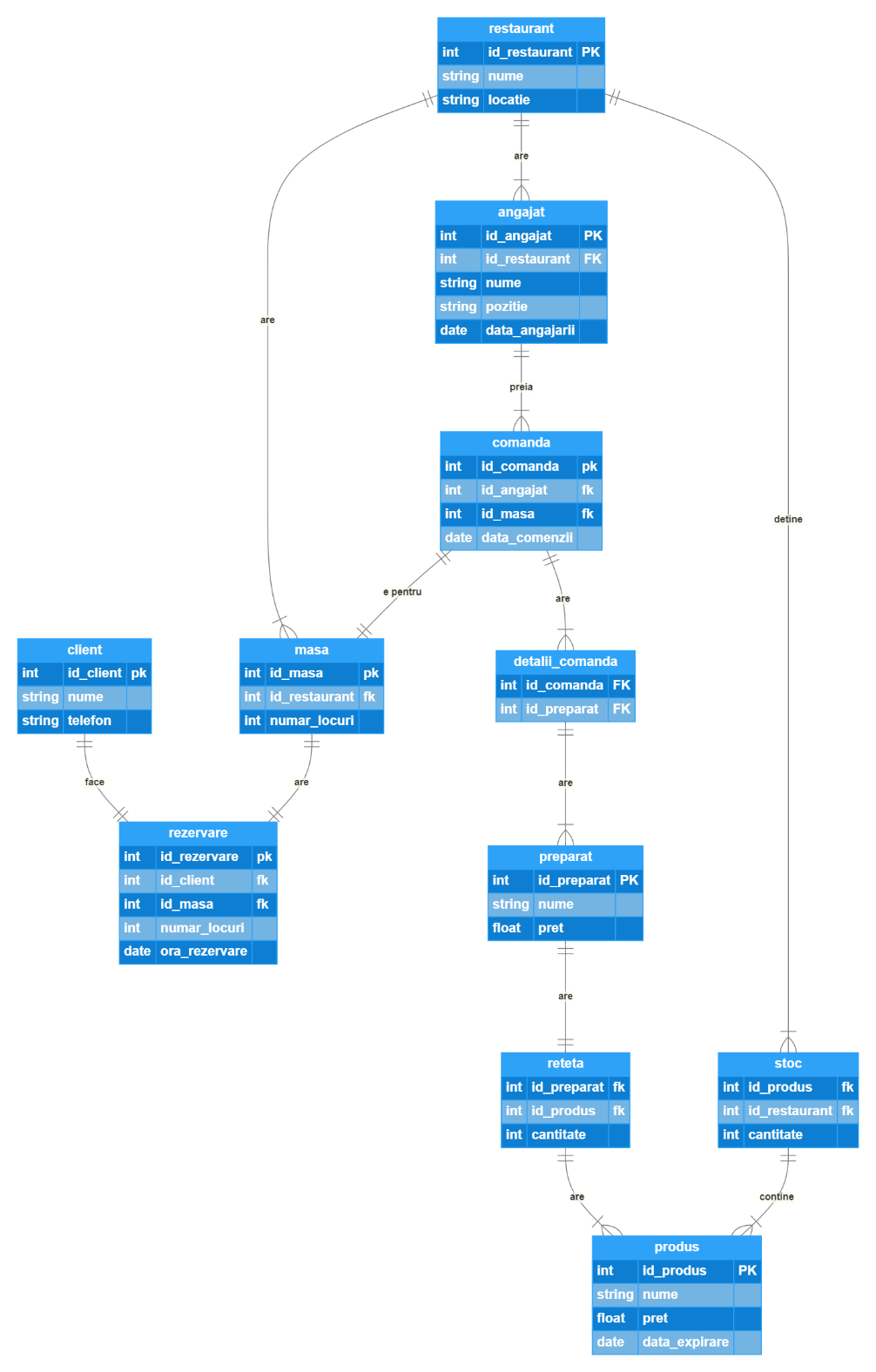
**1.**

Utilitatea unei baze de date a unui lant de restaurante constă în capacitatea sa de a stoca și gestiona datele relevante pentru activitatea restaurantelor, permițând managerilor să ia decizii informate și să optimizeze operațiunile. Baza de date ar putea fi utilizată pentru a optimiza gestionarea stocurilor și menținerea consistenței în ceea ce privește calitatea și gustul mâncării. De asemenea, bază de date poate fi utilizată pentru a monitoriza vânzările și performanța angajaților, permițând managerilor să ia decizii informate privind resursele necesare și investițiile în dezvoltarea afacerii.

**2.**

****

**3.**

****

**4.**

-- Tabela 'restaurant'  
CREATE TABLE restaurant (  
 id\_restaurant INT PRIMARY KEY,  
 nume VARCHAR2(100),  
 locatie VARCHAR2(100)  
);  
  
-- Tabela 'angajat'  
CREATE TABLE angajat (  
 id\_angajat INT PRIMARY KEY,  
 id\_restaurant INT,  
 nume VARCHAR2(100),  
 pozitie VARCHAR2(100),  
 data\_angajarii DATE,  
 FOREIGN KEY (id\_restaurant) REFERENCES restaurant(id\_restaurant)  
);  
  
-- Tabela 'comanda'  
CREATE TABLE comanda (  
 id\_comanda INT PRIMARY KEY,  
 id\_angajat INT,  
 id\_masa INT,  
 data\_comenzii DATE,  
 FOREIGN KEY (id\_angajat) REFERENCES angajat(id\_angajat),  
 FOREIGN KEY (id\_masa) REFERENCES masa(id\_masa)  
);  
  
-- Tabela 'masa'  
CREATE TABLE masa (  
 id\_masa INT PRIMARY KEY,  
 id\_restaurant INT,  
 numar\_locuri INT,  
 FOREIGN KEY (id\_restaurant) REFERENCES restaurant(id\_restaurant)  
);  
  
-- Tabela 'client'  
CREATE TABLE client (  
 id\_client INT PRIMARY KEY,  
 nume VARCHAR2(100),  
 telefon VARCHAR2(15)  
);  
  
-- Tabela 'rezervare'  
CREATE TABLE rezervare (  
 id\_rezervare INT PRIMARY KEY,  
 id\_client INT,  
 id\_masa INT,  
 numar\_locuri INT,  
 ora\_rezervare DATE,  
 FOREIGN KEY (id\_client) REFERENCES client(id\_client),  
 FOREIGN KEY (id\_masa) REFERENCES masa(id\_masa)  
);  
  
-- Tabela 'detalii\_comanda'  
CREATE TABLE detalii\_comanda (  
 id\_comanda INT,  
 id\_preparat INT,  
 FOREIGN KEY (id\_comanda) REFERENCES comanda(id\_comanda),  
 FOREIGN KEY (id\_preparat) REFERENCES preparat(id\_preparat)  
);  
  
-- Tabela 'preparat'  
CREATE TABLE preparat (  
 id\_preparat INT PRIMARY KEY,  
 nume VARCHAR2(100),  
 pret FLOAT  
);  
  
-- Tabela 'reteta'  
CREATE TABLE reteta (  
 id\_preparat INT,  
 id\_produs INT,  
 cantitate INT,  
 FOREIGN KEY (id\_preparat) REFERENCES preparat(id\_preparat),  
 FOREIGN KEY (id\_produs) REFERENCES produs(id\_produs)  
);  
  
-- Tabela 'stoc'  
CREATE TABLE stoc (  
 id\_produs INT,  
 id\_restaurant INT,  
 cantitate INT,  
 FOREIGN KEY (id\_produs) REFERENCES produs(id\_produs),  
 FOREIGN KEY (id\_restaurant) REFERENCES restaurant(id\_restaurant)  
);  
  
-- Tabela 'produs'  
CREATE TABLE produs (  
 id\_produs INT PRIMARY KEY,  
 nume VARCHAR2(100),  
 pret FLOAT,  
 data\_expirare DATE  
);  
  
ALTER TABLE restaurant ADD UNIQUE (id\_restaurant, nume);  
ALTER TABLE angajat ADD UNIQUE (id\_restaurant, id\_angajat);  
ALTER TABLE client ADD UNIQUE (id\_client);  
ALTER TABLE masa ADD UNIQUE (id\_restaurant, id\_masa);  
ALTER TABLE rezervare ADD UNIQUE (id\_client, id\_masa, ora\_rezervare);  
ALTER TABLE detalii\_comanda ADD UNIQUE (id\_comanda, id\_preparat);  
ALTER TABLE reteta ADD UNIQUE (id\_preparat, id\_produs);  
ALTER TABLE stoc ADD UNIQUE (id\_restaurant, id\_produs);  
ALTER TABLE produs ADD UNIQUE (id\_produs);

**5.**

-- Inserarea datelor în tabela 'restaurant'  
INSERT INTO restaurant VALUES (1, 'La Mare', 'Strada Oceanelor 10');  
INSERT INTO restaurant VALUES (2, 'La Pădure', 'Bulevardul Copacilor 5');  
INSERT INTO restaurant VALUES (3, 'Urban Grill', 'Aleea Grătarului 3');  
INSERT INTO restaurant VALUES (4, 'Casa Bunicii', 'Strada Amintirilor 8');  
INSERT INTO restaurant VALUES (5, 'Bistro Moderne', 'Strada Inovației 1');  
  
-- Inserarea datelor în tabela 'angajat'  
INSERT INTO angajat VALUES (1, 1, 'Ion Popescu', 'Chelner',*sysdate*);  
INSERT INTO angajat VALUES (2, 1, 'Ana Maria', 'Bucătar', *TO\_DATE*('2018-02-08', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (3, 1, 'Cristi Serban', 'Manager', *TO\_DATE*('2016-03-09', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (4, 1, 'Maria Pop', 'Chelner', *TO\_DATE*('2019-04-10', 'YYYY-MM-DD'));  
  
INSERT INTO angajat VALUES (5, 2, 'Vlad Ionescu', 'Chelner', *TO\_DATE*('2017-01-11', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (6, 2, 'George Vasile', 'Manager', *TO\_DATE*('2015-06-22', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (7, 2, 'Andreea Popescu', 'Bucătar', *TO\_DATE*('2019-04-03', 'YYYY-MM-DD'));  
  
INSERT INTO angajat VALUES (8, 3, 'Larisa Preda', 'Bucătar', *TO\_DATE*('2018-10-24', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (9, 3, 'Mihai Anton', 'Chelner', *TO\_DATE*('2016-09-15', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (10, 3, 'Andrei Tudose', 'Manager', *TO\_DATE*('2016-08-06', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (11, 3, 'Ecaterina Vasiliu', 'Chelner', *TO\_DATE*('2017-07-27', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (12, 3, 'Alexandru Popica', 'Chelner', *TO\_DATE*('2019-06-18', 'YYYY-MM-DD'));  
  
INSERT INTO angajat VALUES (13, 4, 'Mihnea Irineu', 'Manager', *TO\_DATE*('2015-05-29', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (14, 4, 'Andreea Avram', 'Bucatar', *TO\_DATE*('2016-04-20', 'YYYY-MM-DD'));  
  
INSERT INTO angajat VALUES (15, 5, 'Andrei Munteanu', 'Chelner', *TO\_DATE*('2019-02-21', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (16, 5, 'Ioana Manea', 'Bucatar', *TO\_DATE*('2018-11-02', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (17, 5, 'Bogdan Dinu', 'Manager', *TO\_DATE*('2017-4-01', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (18, 5, 'Laura Ciubotaru', 'Chelner', *TO\_DATE*('2016-12-10', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (19, 5, 'Dan Marinescu', 'Chelner', *TO\_DATE*('2022-09-15', 'YYYY-MM-DD'));  
INSERT INTO angajat VALUES (20, 5, 'Raluca Bulcea', 'Chelner', *TO\_DATE*('2020-09-24', 'YYYY-MM-DD'));  
  
  
-- Inserarea datelor în tabela 'masa'  
INSERT INTO masa VALUES (1, 1, 4);  
INSERT INTO masa VALUES (2, 1, 2);  
INSERT INTO masa VALUES (3, 1, 2);  
INSERT INTO masa VALUES (4, 1, 6);  
  
INSERT INTO masa VALUES (5, 2, 6);  
INSERT INTO masa VALUES (6, 2, 8);  
INSERT INTO masa VALUES (7, 2, 4);  
INSERT INTO masa VALUES (8, 2, 2);  
  
INSERT INTO masa VALUES (9, 3, 4);  
INSERT INTO masa VALUES (10, 3, 2);  
INSERT INTO masa VALUES (11, 3, 2);  
INSERT INTO masa VALUES (12, 3, 6);  
INSERT INTO masa VALUES (13, 3, 6);  
INSERT INTO masa VALUES (14, 3, 8);  
INSERT INTO masa VALUES (15, 3, 4);  
INSERT INTO masa VALUES (16, 3, 4);  
  
INSERT INTO masa VALUES (17, 4, 4);  
INSERT INTO masa VALUES (18, 4, 2);  
INSERT INTO masa VALUES (19, 4, 2);  
  
INSERT INTO masa VALUES (20, 5, 4);  
INSERT INTO masa VALUES (21, 5, 2);  
INSERT INTO masa VALUES (22, 5, 2);  
INSERT INTO masa VALUES (23, 5, 6);  
INSERT INTO masa VALUES (24, 5, 6);  
INSERT INTO masa VALUES (25, 5, 8);  
INSERT INTO masa VALUES (26, 5, 4);  
  
  
-- Inserarea datelor în tabela 'client'  
INSERT INTO client VALUES (1, 'Andrei Muntea', '0722123456');  
INSERT INTO client VALUES (2, 'Ioana Grigoras', '0733344567');  
INSERT INTO client VALUES (3, 'Bogdan Raducan', '0744455678');  
INSERT INTO client VALUES (4, 'Laura Popoiu', '0755566789');  
INSERT INTO client VALUES (5, 'Dan Carturasu', '0766677890');  
INSERT INTO client VALUES (6, 'Cătălina Sorescu', '0777888901');  
INSERT INTO client VALUES (7, 'Florin Călinescu', '0777888902');  
INSERT INTO client VALUES (8, 'Diana Dumitrescu', '0777888903');  
INSERT INTO client VALUES (9, 'Bogdan Stan', '0777888904');  
INSERT INTO client VALUES (10, 'Alina Manea', '0777888905');  
INSERT INTO client VALUES (11, 'Teodor Ionescu', '0777888906');  
INSERT INTO client VALUES (12, 'Sorina Muntean', '0777888907');  
INSERT INTO client VALUES (13, 'Cristian Popescu', '0777888908');  
INSERT INTO client VALUES (14, 'Monica Anghel', '0777888909');  
INSERT INTO client VALUES (15, 'Radu Toma', '0777888910');  
  
  
-- Inserarea datelor în tabela 'rezervare'  
INSERT INTO rezervare VALUES (1, 2, 3, 2, *TO\_DATE*('2024-02-18 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (2, 5, 1, 4, *TO\_DATE*('2024-02-19 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (3, 3, 8, 2, *TO\_DATE*('2024-02-20 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (4, 6, 12, 6, *TO\_DATE*('2024-02-21 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (5, 4, 18, 2, *TO\_DATE*('2024-02-22 19:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (6, 10, 5, 6, *TO\_DATE*('2024-02-23 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (7, 9, 26, 4, *TO\_DATE*('2024-02-24 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (8, 7, 15, 4, *TO\_DATE*('2024-02-25 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (9, 8, 21, 2, *TO\_DATE*('2024-02-26 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (10, 15, 9, 4, *TO\_DATE*('2024-02-27 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (11, 11, 2, 2, *TO\_DATE*('2024-02-28 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (12, 14, 14, 8, *TO\_DATE*('2024-03-01 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (13, 13, 17, 4, *TO\_DATE*('2024-03-02 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (14, 12, 6, 8, *TO\_DATE*('2024-03-03 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (15, 12, 22, 2, *TO\_DATE*('2024-03-04 19:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (16, 11, 24, 6, *TO\_DATE*('2024-03-05 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (17, 14, 10, 2, *TO\_DATE*('2024-03-06 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (18, 13, 25, 8, *TO\_DATE*('2024-03-07 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (19, 15, 13, 6, *TO\_DATE*('2024-03-08 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (20, 10, 20, 4, *TO\_DATE*('2024-03-09 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (21, 9, 16, 4, *TO\_DATE*('2024-03-10 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (22, 8, 7, 4, *TO\_DATE*('2024-03-11 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (23, 7, 23, 6, *TO\_DATE*('2024-03-12 19:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (24, 6, 19, 2, *TO\_DATE*('2024-03-13 20:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO rezervare VALUES (25, 5, 11, 2, *TO\_DATE*('2024-03-14 18:00', 'YYYY-MM-DD HH24:MI'));  
  
  
-- Inserarea datelor în tabela 'preparat'  
INSERT INTO preparat VALUES (1, 'Supă de legume', 10.0);  
INSERT INTO preparat VALUES (2, 'Friptură de vită', 50.0);  
INSERT INTO preparat VALUES (3, 'Salată grecească', 20.0);  
INSERT INTO preparat VALUES (4, 'Peste la grătar', 45.0);  
INSERT INTO preparat VALUES (5, 'Pizza Margherita', 30.0);  
INSERT INTO preparat VALUES (6, 'Spaghetti Carbonara', 35.0);  
INSERT INTO preparat VALUES (7, 'Risotto cu ciuperci', 25.0);  
INSERT INTO preparat VALUES (8, 'Escalop de porc', 40.0);  
INSERT INTO preparat VALUES (9, 'Salată de quinoa cu legume', 22.0);  
INSERT INTO preparat VALUES (10, 'Burger de somon', 55.0);  
  
-- Inserarea datelor în tabela 'produs'  
INSERT INTO produs VALUES (1, 'Faina', 3.5, *TO\_DATE*('2025-06-15', 'YYYY-MM-DD'));  
INSERT INTO produs VALUES (2, 'Pui', 1.2, *TO\_DATE*('2024-02-10', 'YYYY-MM-DD'));  
INSERT INTO produs VALUES (3, 'Porc', 2.5, *TO\_DATE*('2024-02-20', 'YYYY-MM-DD'));  
INSERT INTO produs VALUES (4, 'Ouă', 0.8, *TO\_DATE*('2024-02-05', 'YYYY-MM-DD'));  
INSERT INTO produs VALUES (5, 'Peste', 4.5, *TO\_DATE*('2024-02-22', 'YYYY-MM-DD'));  
INSERT INTO produs VALUES (6, 'Roșii', 2.3, *TO\_DATE*('2024-03-18', 'YYYY-MM-DD'));  
  
  
-- Inserarea datelor în tabela 'comanda'  
INSERT INTO comanda VALUES (1, 1, 3, *TO\_DATE*('2024-01-06 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (2, 4, 3, *TO\_DATE*('2024-01-06 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (3, 5, 7, *TO\_DATE*('2024-01-07 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (4, 4, 2, *TO\_DATE*('2024-01-08 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (5, 1, 2, *TO\_DATE*('2024-01-08 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (6, 9, 10, *TO\_DATE*('2024-01-09 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (7, 18, 23, *TO\_DATE*('2024-01-10 18:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (8, 12, 11, *TO\_DATE*('2024-01-11 18:30', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (9, 19, 25, *TO\_DATE*('2024-01-12 19:00', 'YYYY-MM-DD HH24:MI'));  
INSERT INTO comanda VALUES (10, 11, 14, *TO\_DATE*('2024-01-12 19:30', 'YYYY-MM-DD HH24:MI'));  
  
-- Inserarea datelor în tabela 'detalii\_comanda'  
INSERT INTO detalii\_comanda VALUES (1, 1);  
INSERT INTO detalii\_comanda VALUES (1, 2);  
INSERT INTO detalii\_comanda VALUES (2, 3);  
INSERT INTO detalii\_comanda VALUES (2, 4);  
INSERT INTO detalii\_comanda VALUES (3, 5);  
INSERT INTO detalii\_comanda VALUES (4, 6);  
INSERT INTO detalii\_comanda VALUES (5, 7);  
INSERT INTO detalii\_comanda VALUES (6, 8);  
INSERT INTO detalii\_comanda VALUES (7, 9);  
INSERT INTO detalii\_comanda VALUES (8, 10);  
INSERT INTO detalii\_comanda VALUES (9, 1);  
INSERT INTO detalii\_comanda VALUES (10, 2);  
  
select \* from detalii\_comanda;  
  
-- Inserarea datelor în tabela 'reteta'  
INSERT INTO reteta VALUES (1, 1, 2);  
INSERT INTO reteta VALUES (1, 2, 1);  
  
INSERT INTO reteta VALUES (2, 3, 1);  
INSERT INTO reteta VALUES (2, 4, 2);  
  
INSERT INTO reteta VALUES (3, 5, 1);  
  
INSERT INTO reteta VALUES (4, 2, 1);  
INSERT INTO reteta VALUES (4, 3, 1);  
INSERT INTO reteta VALUES (4, 4, 1);  
INSERT INTO reteta VALUES (4, 5, 1);  
  
INSERT INTO reteta VALUES (5, 5, 2);  
INSERT INTO reteta VALUES (5, 1, 1);  
INSERT INTO reteta VALUES (5, 2, 1);  
  
INSERT INTO reteta VALUES (6, 1, 1);  
INSERT INTO reteta VALUES (6, 3, 1);  
  
INSERT INTO reteta VALUES (7, 2, 1);  
INSERT INTO reteta VALUES (7, 4, 1);  
INSERT INTO reteta VALUES (7, 5, 1);  
  
INSERT INTO reteta VALUES (8, 5, 1);  
INSERT INTO reteta VALUES (8, 2, 1);  
  
INSERT INTO reteta VALUES (9, 3, 2);  
INSERT INTO reteta VALUES (9, 1, 1);  
  
INSERT INTO reteta VALUES (10, 4, 1);  
  
  
-- Inserarea datelor în tabela 'stoc'  
INSERT INTO stoc VALUES (1, 1, 50);  
INSERT INTO stoc VALUES (1, 2, 60);  
INSERT INTO stoc VALUES (1, 3, 55);  
INSERT INTO stoc VALUES (1, 4, 45);  
INSERT INTO stoc VALUES (1, 5, 65);  
  
INSERT INTO stoc VALUES (2, 1, 100);  
INSERT INTO stoc VALUES (2, 2, 120);  
INSERT INTO stoc VALUES (2, 3, 110);  
INSERT INTO stoc VALUES (2, 4, 90);  
INSERT INTO stoc VALUES (2, 5, 130);  
  
INSERT INTO stoc VALUES (3, 1, 70);  
INSERT INTO stoc VALUES (3, 2, 80);  
INSERT INTO stoc VALUES (3, 3, 75);  
INSERT INTO stoc VALUES (3, 4, 65);  
INSERT INTO stoc VALUES (3, 5, 85);  
  
INSERT INTO stoc VALUES (4, 1, 200);  
INSERT INTO stoc VALUES (4, 2, 220);  
INSERT INTO stoc VALUES (4, 3, 210);  
INSERT INTO stoc VALUES (4, 4, 90);  
INSERT INTO stoc VALUES (4, 5, 230);  
  
INSERT INTO stoc VALUES (5, 1, 40);  
INSERT INTO stoc VALUES (5, 2, 50);  
INSERT INTO stoc VALUES (5, 3, 45);  
INSERT INTO stoc VALUES (5, 4, 35);  
INSERT INTO stoc VALUES (5, 5, 55);  
  
INSERT INTO stoc VALUES (6, 1, 60);  
INSERT INTO stoc VALUES (6, 2, 70);  
INSERT INTO stoc VALUES (6, 3, 65);  
INSERT INTO stoc VALUES (6, 4, 55);  
INSERT INTO stoc VALUES (6, 5, 75);

**6.**

**Creez o procedura care calculeaza top 5 cele mai vandute preparate, in functie de comenzi.**

CREATE OR REPLACE PROCEDURE analizeaza\_preferinte IS

-- Tipuri de colectii

TYPE FrecventaPreparateType IS TABLE OF NUMBER INDEX BY VARCHAR2(100);

TYPE TabelPreparateType IS TABLE OF VARCHAR2(100);

TYPE TopPreparateType IS VARRAY(5) OF VARCHAR2(100);

-- Colectii pentru fiecare tip de date

frecventaPreparate FrecventaPreparateType;

listaPreparate TabelPreparateType;

topPreparate TopPreparateType := TopPreparateType();

-- Variabile auxiliare

numePreparat VARCHAR2(100);

maxFrecventa NUMBER;

keyMaxFrecventa VARCHAR2(100);

-- Functie interna pentru a gasi preparatul cu frecventa maxima

FUNCTION gasire\_preparat\_maxim RETURN VARCHAR2 IS

tempKey VARCHAR2(100);

maxValue NUMBER := 0;

BEGIN

tempKey := frecventaPreparate.FIRST;

WHILE tempKey IS NOT NULL LOOP

IF frecventaPreparate(tempKey) > maxValue THEN

maxValue := frecventaPreparate(tempKey);

keyMaxFrecventa := tempKey;

END IF;

tempKey := frecventaPreparate.NEXT(tempKey);

END LOOP;

RETURN keyMaxFrecventa;

END;

BEGIN

-- Pasul 1: Construim frecventa preparatelor comandate

FOR recDetaliuComanda IN (SELECT dc.id\_preparat, p.nume

FROM detalii\_comanda dc

JOIN preparat p ON dc.id\_preparat = p.id\_preparat) LOOP

IF frecventaPreparate.EXISTS(recDetaliuComanda.nume) THEN

frecventaPreparate(recDetaliuComanda.nume) := frecventaPreparate(recDetaliuComanda.nume) + 1;

ELSE

frecventaPreparate(recDetaliuComanda.nume) := 1;

END IF;

END LOOP;

-- Pasul 2: Selectam cele mai comandate 5 preparate

FOR i IN 1..5 LOOP

keyMaxFrecventa := gasire\_preparat\_maxim();

EXIT WHEN keyMaxFrecventa IS NULL; -- Iesim daca nu mai sunt preparate

-- Adaugam preparatul in VARRAY

topPreparate.EXTEND;

topPreparate(i) := keyMaxFrecventa;

-- Eliminam preparatul actual pentru a gasi urmatorul cel mai frecvent comandat

frecventaPreparate.DELETE(keyMaxFrecventa);

END LOOP;

-- Afisam rezultatul

DBMS\_OUTPUT.PUT\_LINE('Top Preparate:');

FOR i IN 1..topPreparate.COUNT LOOP

DBMS\_OUTPUT.PUT\_LINE(i || '. ' || topPreparate(i));

END LOOP;

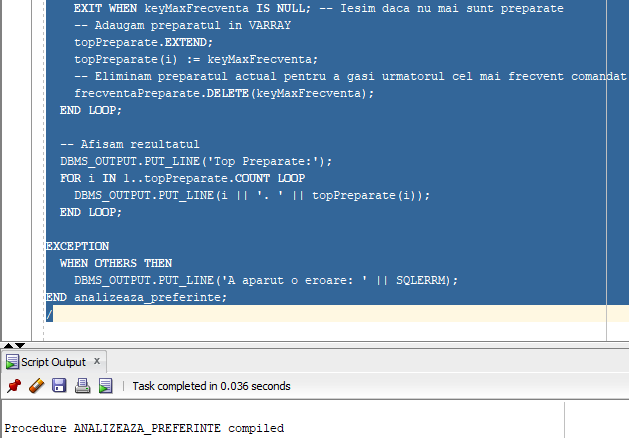
EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o eroare: ' || SQLERRM);

END analizeaza\_preferinte;

/

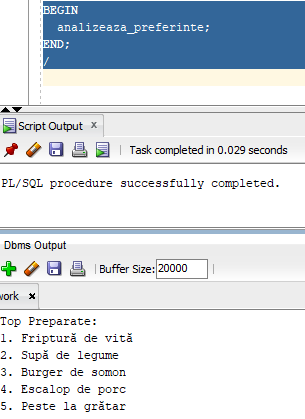
****

BEGIN

analizeaza\_preferinte;

END;

/

****

**7.**

**Creez o procedura care afiseaza detaliile unei comenzi dintr-o anumita data.**

CREATE OR REPLACE PROCEDURE raport\_zilnic (p\_report\_date DATE)

IS

-- Definirea cursorului neparametrizat pentru toate comenzile

CURSOR c\_orders IS

SELECT id\_comanda, id\_angajat, id\_masa, data\_comenzii

FROM comanda

WHERE data\_comenzii = p\_report\_date;

-- Definirea cursorului parametrizat pentru detalii comanda, dependent de comanda selectata

CURSOR c\_order\_details (c\_comanda\_id comanda.id\_comanda%TYPE) IS

SELECT dc.id\_comanda, p.nume, p.pret

FROM detalii\_comanda dc

JOIN preparat p ON dc.id\_preparat = p.id\_preparat

WHERE dc.id\_comanda = c\_comanda\_id;

-- Variabila pentru stocarea inregistrarii curente a comenzii

v\_order c\_orders%ROWTYPE;

-- Variabila pentru stocarea inregistrarii curente a detaliilor comenzii

v\_order\_detail c\_order\_details%ROWTYPE;

BEGIN

-- Deschiderea cursorului neparametrizat

OPEN c\_orders;

-- Fetch in cursorul comenzilor

LOOP

FETCH c\_orders INTO v\_order;

EXIT WHEN c\_orders%NOTFOUND;

-- Afisarea informatiilor comenzii

DBMS\_OUTPUT.PUT\_LINE('Comanda ID: ' || v\_order.id\_comanda ||

', Angajat ID: ' || v\_order.id\_angajat ||

', Masa ID: ' || v\_order.id\_masa ||

', Data Comenzii: ' || v\_order.data\_comenzii);

-- Deschiderea cursorului parametrizat pentru detalii comanda

OPEN c\_order\_details(v\_order.id\_comanda);

-- Fetch in cursorul detaliilor comenzilor

LOOP

FETCH c\_order\_details INTO v\_order\_detail;

EXIT WHEN c\_order\_details%NOTFOUND;

-- Afisarea detaliilor comenzii

DBMS\_OUTPUT.PUT\_LINE(' Preparat: ' || v\_order\_detail.nume ||

', Pret: ' || v\_order\_detail.pret);

END LOOP;

-- inchiderea cursorului pentru detalii comanda

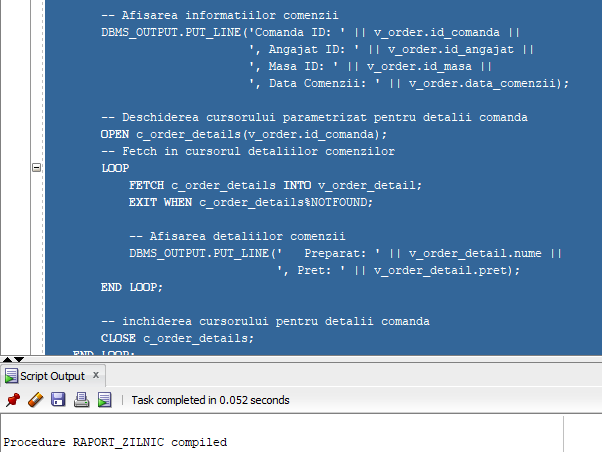
CLOSE c\_order\_details;

END LOOP;

-- inchiderea cursorului comenzilor

CLOSE c\_orders;

END raport\_zilnic;

/****

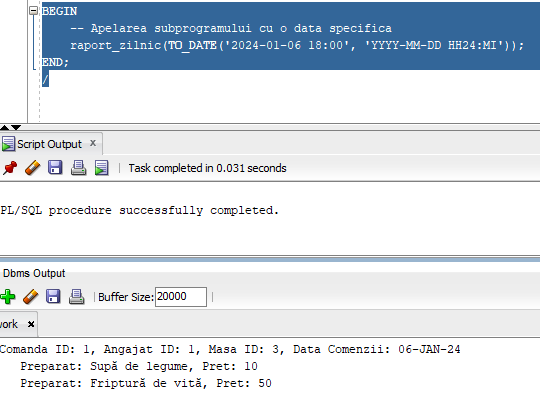
BEGIN

-- Apelarea subprogramului cu o data specifica

raport\_zilnic(TO\_DATE('2024-01-06 18:00', 'YYYY-MM-DD HH24:MI'));

END;

/

****

**8.**

**Sa se calculeze venitul total generat de un angajat intr-o anumita zi. Exceptiile pe care le propun pentru aceasta problema sunt: angajatul nu exista in baza de date, angajatul nu are vanzari in ziua respectiva.**

CREATE OR REPLACE FUNCTION venit\_angajat\_zilnic (p\_id\_angajat INT, p\_data DATE)

RETURN NUMBER

IS

total\_venit NUMBER := 0;

angajat\_negasit EXCEPTION;

fara\_vanzari EXCEPTION;

v\_angajat\_count INT;

BEGIN

-- Verificam daca angajatul exista

SELECT COUNT(\*)

INTO v\_angajat\_count

FROM angajat

WHERE id\_angajat = p\_id\_angajat;

IF v\_angajat\_count = 0 THEN

RAISE angajat\_negasit;

END IF;

-- Calculam venitul total generat de angajat din comenzi in ziua specificata

SELECT NVL(SUM(p.pret), 0)

INTO total\_venit

FROM comanda c

INNER JOIN detalii\_comanda dc ON c.id\_comanda = dc.id\_comanda

INNER JOIN preparat p ON dc.id\_preparat = p.id\_preparat

WHERE c.id\_angajat = p\_id\_angajat AND TRUNC(c.data\_comenzii) = TRUNC(p\_data);

IF total\_venit = 0 THEN

RAISE fara\_vanzari;

END IF;

RETURN total\_venit;

EXCEPTION

WHEN angajat\_negasit THEN

DBMS\_OUTPUT.PUT\_LINE('Angajatul cu ID-ul ' || p\_id\_angajat || ' nu a fost gasit.');

RETURN NULL;

WHEN fara\_vanzari THEN

DBMS\_OUTPUT.PUT\_LINE('Angajatul cu ID-ul ' || p\_id\_angajat || ' nu are vanzari in data de ' || TO\_CHAR(p\_data, 'DD-MM-YYYY') || '.');

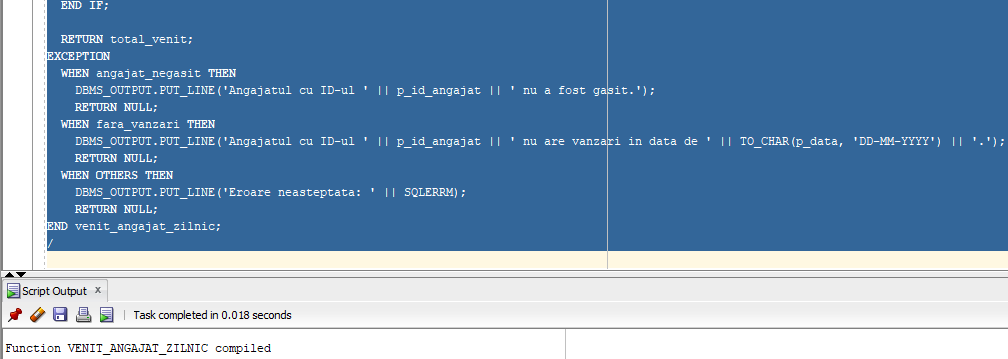
RETURN NULL;

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare neasteptata: ' || SQLERRM);

RETURN NULL;

END venit\_angajat\_zilnic;

/****

**Apeluri subprogram:**

DECLARE

venit\_total NUMBER;

BEGIN

venit\_total := venit\_angajat\_zilnic(1, TO\_DATE('2024-01-06', 'YYYY-MM-DD'));

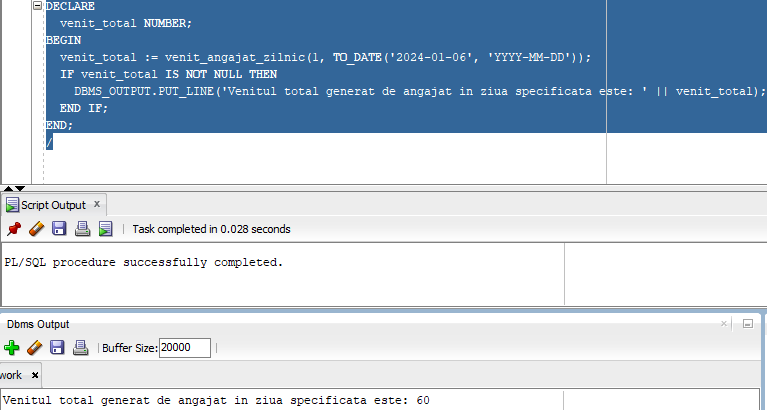
IF venit\_total IS NOT NULL THEN

DBMS\_OUTPUT.PUT\_LINE('Venitul total generat de angajat in ziua specificata este: ' || venit\_total);

END IF;

END;

/



DECLARE

venit\_total NUMBER;

BEGIN

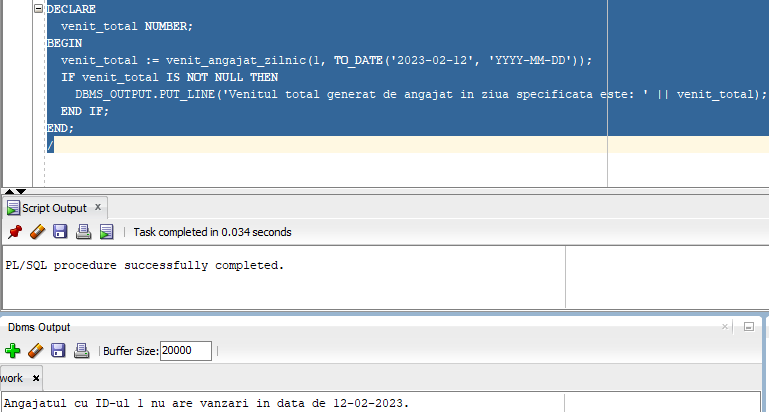
venit\_total := venit\_angajat\_zilnic(1, TO\_DATE('2023-02-12', 'YYYY-MM-DD'));

IF venit\_total IS NOT NULL THEN

DBMS\_OUTPUT.PUT\_LINE('Venitul total generat de angajat in ziua specificata este: ' || venit\_total);

END IF;

END;

/ 

DECLARE

venit\_total NUMBER;

BEGIN

venit\_total := venit\_angajat\_zilnic(25, TO\_DATE('2024-01-06', 'YYYY-MM-DD'));

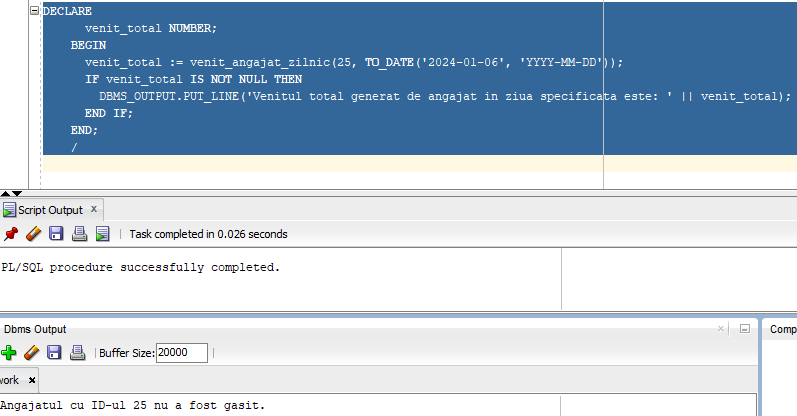
IF venit\_total IS NOT NULL THEN

DBMS\_OUTPUT.PUT\_LINE('Venitul total generat de angajat in ziua specificata este: ' || venit\_total);

END IF;

END;

/



**9.**

**Creez un subprogram stocat independent de tip procedura care modifica cantitatea stocului produselor in functie de comenzi. Comenzile considerate valide sunt cele formate dintr-un singur preparat. Pentru cele formate din 2 sau mai multe preparate se arunca exceptia TOO\_MANY\_ROWS.**

CREATE OR REPLACE PROCEDURE actualizare\_stoc(idComanda IN INT) IS

cantitate\_necesara NUMBER;

cantitate\_stoc NUMBER;

idProdus NUMBER;

idRestaurant NUMBER;

idPreparat NUMBER;

BEGIN

-- Obtinem id-ul restaurantului prin angajatul care a preluat comanda

SELECT a.id\_restaurant INTO idRestaurant

FROM comanda c

JOIN angajat a ON c.id\_angajat = a.id\_angajat

WHERE c.id\_comanda = idComanda;

-- Incercam sa obtinem un singur preparat din comanda, asteptandu-ne la un singur rezultat

SELECT id\_preparat INTO idPreparat

FROM detalii\_comanda

WHERE id\_comanda = idComanda;

-- Continuam cu logica de actualizare a stocului doar pentru acel preparat

SELECT r.cantitate, r.id\_produs INTO cantitate\_necesara, idProdus

FROM reteta r

WHERE r.id\_preparat = idPreparat;

-- Verificam stocul pentru produsul respectiv

SELECT s.cantitate INTO cantitate\_stoc

FROM stoc s

WHERE s.id\_produs = idProdus AND s.id\_restaurant = idRestaurant;

IF cantitate\_stoc < cantitate\_necesara THEN

-- Daca stocul nu este suficient, aruncam o exceptie

RAISE\_APPLICATION\_ERROR(-20002, 'Stoc insuficient pentru produsul cu ID-ul ' || idProdus);

ELSE

-- Actualizam stocul

UPDATE stoc SET cantitate = cantitate\_stoc - cantitate\_necesara

WHERE id\_produs = idProdus AND id\_restaurant = idRestaurant;

END IF;

-- Salvam modificarile

COMMIT;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

-- Tratam cazul in care o cautare nu returneaza date

ROLLBACK;

RAISE\_APPLICATION\_ERROR(-20003, 'Datele necesare nu au fost gasite pentru comanda cu ID-ul ' || idComanda);

WHEN OTHERS THEN

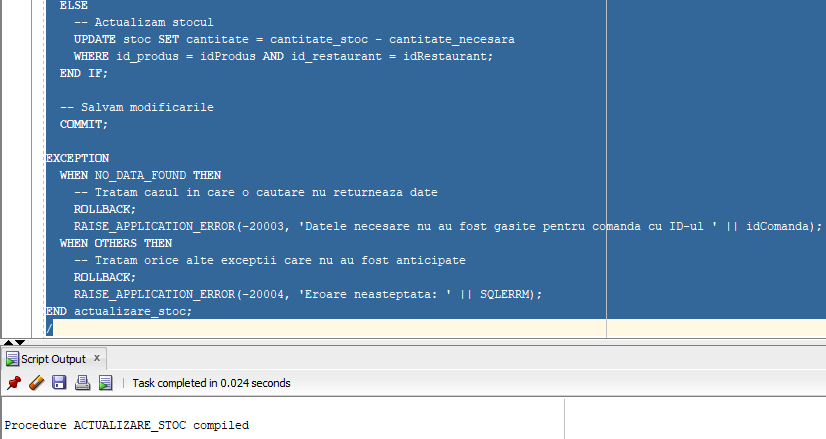
-- Tratam orice alte exceptii care nu au fost anticipate

ROLLBACK;

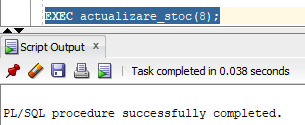
RAISE\_APPLICATION\_ERROR(-20004, 'Eroare neasteptata: ' || SQLERRM);

END actualizare\_stoc;

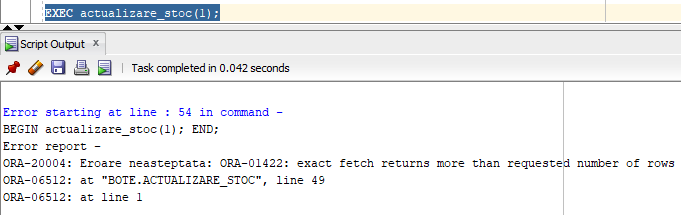
/

****

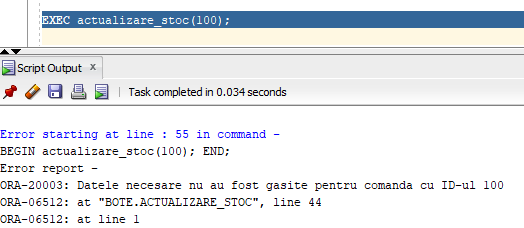
EXEC actualizare\_stoc(8);

****

EXEC actualizare\_stoc(1);



EXEC actualizare\_stoc(100);



**10.**

**Creez un trigger LMD la nivel de comanda care afiseaza la modificarari in tabela angajati cati chelneri sunt angajati.**

CREATE OR REPLACE TRIGGER afisare\_nr\_chelneri

AFTER INSERT OR UPDATE OR DELETE ON angajat

DECLARE

numarChelneri NUMBER;

BEGIN

-- Calculam numarul de chelneri angajati

SELECT COUNT(\*)

INTO numarChelneri

FROM angajat

WHERE pozitie = 'Chelner';

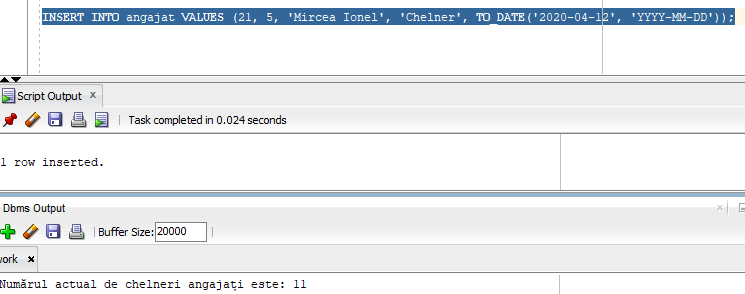
-- Afisam numarul de chelneri

DBMS\_OUTPUT.PUT\_LINE('Numarul actual de chelneri angajati este: ' || numarChelneri);

END;



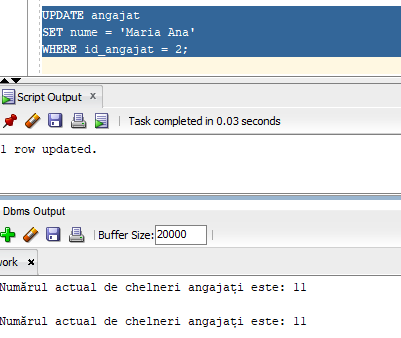
INSERT INTO angajat VALUES (21, 5, 'Mircea Ionel', 'Chelner', TO\_DATE('2020-04-12', 'YYYY-MM-DD'));



UPDATE angajat

SET nume = 'Maria Ana'

WHERE id\_angajat = 2;



**11.**

**Creez un trigger LMD la nivel de linie pentru verificarea capacitatii unei mese in momentul in care este creata o rezervare noua.**

CREATE OR REPLACE TRIGGER verificare\_locuri\_masa

BEFORE INSERT ON rezervare

FOR EACH ROW

DECLARE

numar\_locuri\_disponibile INT;

BEGIN

-- Verifica numarul de locuri de la masa pentru care se face rezervarea

SELECT numar\_locuri INTO numar\_locuri\_disponibile

FROM masa

WHERE id\_masa = :NEW.id\_masa;

-- Verifica daca numarul de locuri solicitate pentru rezervare nu depaseste capacitatea mesei

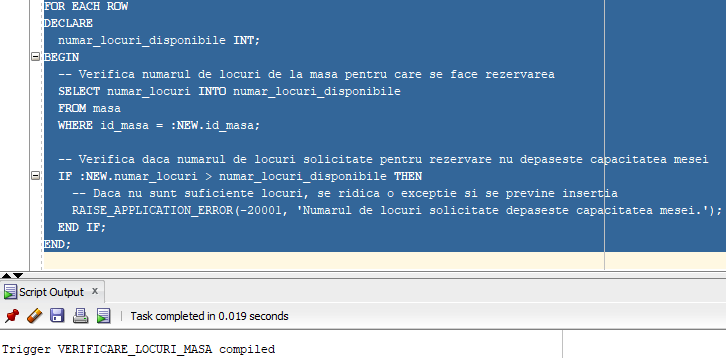
IF :NEW.numar\_locuri > numar\_locuri\_disponibile THEN

-- Daca nu sunt suficiente locuri, se ridica o exceptie si se previne insertia

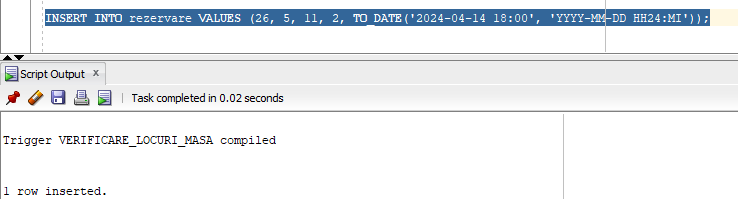
RAISE\_APPLICATION\_ERROR(-20001, 'Numarul de locuri solicitate depaseste capacitatea mesei.');

END IF;

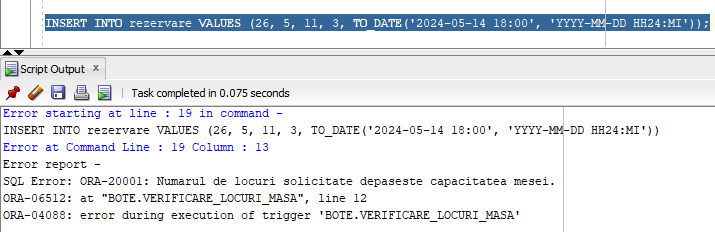
END;



INSERT INTO rezervare VALUES (26, 5, 11, 2, TO\_DATE('2024-04-14 18:00', 'YYYY-MM-DD HH24:MI'));



INSERT INTO rezervare VALUES (26, 5, 11, 3, TO\_DATE('2024-05-14 18:00', 'YYYY-MM-DD HH24:MI'));



**12.**

**LDD care contorizeaza stergerea,modificarea sau crearea de tabele. Singurul user cu drepturi de modifcare este BOTE.**

-- Crearea tabelului de audit

CREATE TABLE audit\_ldd (

id NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY PRIMARY KEY,

action\_type VARCHAR2(30),

object\_type VARCHAR2(30),

object\_name VARCHAR2(128),

username VARCHAR2(30),

timestamp TIMESTAMP

);

-- Crearea trigger-ului LDD

CREATE OR REPLACE TRIGGER audit\_ldd\_trigger

AFTER DROP OR ALTER OR CREATE ON DATABASE

DECLARE

v\_username VARCHAR2(30);

BEGIN

-- Obtinerea numele de utilizator curent

SELECT ora\_login\_user INTO v\_username FROM dual;

-- Verificarea daca utilizatorul este 'BOTE'

IF v\_username = 'BOTE' THEN

-- Inserarea inregistrarii in tabelul de audit

INSERT INTO audit\_ldd(action\_type, object\_type, object\_name, username, timestamp)

VALUES(ora\_sysevent, ora\_dict\_obj\_type, ora\_dict\_obj\_name, v\_username, SYSTIMESTAMP);

ELSE

-- Ridicarea exceptiei daca un alt utilizator incearca sa efectueze LDD

RAISE\_APPLICATION\_ERROR(-20001, 'Nu ai permisiunea de a modifica schema bazei de date.');

END IF;

EXCEPTION

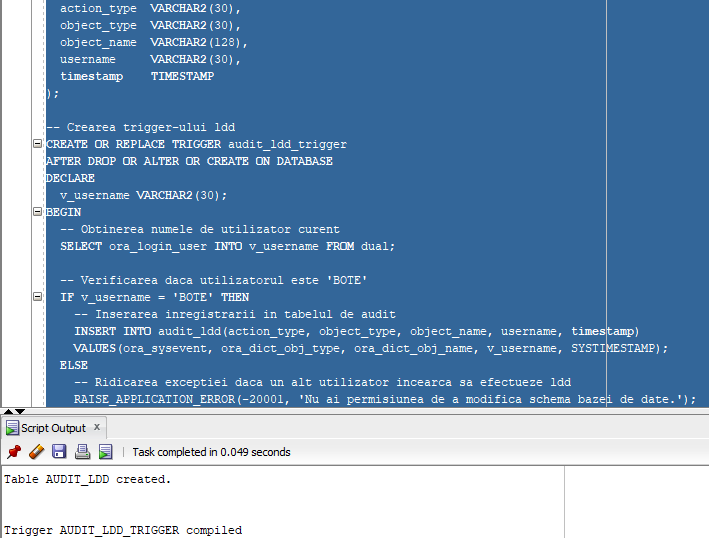
WHEN OTHERS THEN

-- Gestionare exceptiile aici, daca este necesar

NULL;

END;

/



CREATE TABLE meniu (

id\_preparat NUMBER PRIMARY KEY,

nume VARCHAR2(100) NOT NULL,

pret NUMBER(5,2) NOT NULL

);

DROP TABLE meniu;

select\*

from audit\_ldd;

